

**Amendments to the Specification:**

Please replace the paragraph under the heading of "Cross References to Related Applications" on page 1, lines 7 through 11, with the following amended paragraph:

a1  
This application claims priority to provisional application Serial No. 60/154,777 (TI-29686PS), provisional application Serial No. 60/154,657 (TI-29687PS), and provisional application Serial No. 60/154,656 (TI-29688PS), and is related to U.S. Patent 6,546,477 application Serial No. \_\_\_\_\_ (TI-29686) and application Serial No. 09/667,393 \_\_\_\_\_ (TI-29687).

Please replace the paragraph on page 9, lines 18 through 20, with the following amended paragraph:

a2  
Figures 7A, 7B and 7C together ~~is~~ are a code listing illustrating code for an example of an abstract algorithm instance (IALG) interface for the component model of Figure 3;

Please remove the comma after "algMoved()" by replacing the paragraph on page 10, lines 22 through 25, with the following amended paragraph:

a3  
Figure 17 is a code listing of an example implementation of a function algMoved(); that performs any reinitialization necessary to insure that all internal data references are recomputed after a framework relocates an algorithm at run-time in the flow of Figure 8;

Please replace the paragraph on page 14, lines 16 through 24, with the following amended paragraph:

a4  
Access Adapter 107 ~~102~~ is a hardware component that connects development host 106 ~~100~~ to a target system. It utilizes one or more hardware interfaces and/or protocols to convert messages created by user interface commands to the debug command. This component could be a Texas Instruments XDS 510WS controller, for example, connected to the target system debug interface and to the PC through a SCSI port. An alternate

a4 configuration is a Texas Instruments XDS 510 controller installed in a PC connected to the target system debug interface. An example access adapter is described fully in U. S. Patent 5,329,471.

---

Please replace the paragraph that begins on page 14, line 25, and ends on page 15, line 5, with the following amended paragraph:

---

a5 Target System 108 ~~104~~ is comprised of one or more DSPs. The DSP(s) contain hardware designed explicitly to ease the chore of debugging. It is the lowest element of the system debug environment. The DSP debug facilities allow the user to control the program execution, examine the state of the system, memory, and core resources in both real-time and stop mode debug modes. Target System 108 ~~104~~ may be a cellular telephone, for example. Once a software program is written, compiled, linked and loaded onto the target system, it is then debugged and refined. Advantageously, a software program that embodies the present invention can be developed in a manner that minimizes debugging and redesign.

---

Please replace the paragraph on page 15, lines 6 through 9, with the following amended paragraph:

---

a6 By enabling system integrators to "plug-replace" one algorithm for another, the time to market is reduced because the system integrator can choose algorithms from multiple vendors. A huge catalog of inter-operable parts from which any system can be built and can be easily created.

---

Please replace the paragraph on page 24, lines 19 through 29, with the following amended paragraph:

---

a7 Run-time creation of objects is valuable even in systems that do not support or require run-time deletion of these objects. The precise nature of the objects, the number of objects, and even the type of objects created may be a function of parameters that are only available at run-time. For example, a programmer may want to create a single program that works in a variety of hardware platforms that differ in the amount of memory available and

a7  
the amount ~~mount~~ is determinable at run-time. Note that the algorithms conforming to an embodiment of the present invention are modules of a special type, which are referred to as algorithm modules herein. How the algorithm modules support run-time object creation is another embodiment of the present invention described later in this specification.

---

Please replace the paragraph on page 25, lines 14 through 24, with the following amended paragraph:

---

a8  
Suppose, for example, that a module that implements digital filters exists. There are several special cases for digital filters that have significant performance differences such as as ~~ass~~ all-pole, all-zero, and pole-zero filters. Moreover, for certain DSP architectures if one assumes that the filter's data buffers are aligned on certain boundaries, the implementation can take advantage of special data addressing modes and significantly reduce the time required to complete the computation. A filter module may include a global configuration parameter that specifies that the system will only use all-zero filters with aligned data. By making this a design-time global configuration parameter, systems that are willing to accept constraints in their use of the API are rewarded by faster operation of the module that implements the API.

---

Please replace the paragraph on page 49, lines 6 through 7, with the following amended paragraph:

---

a9  
The first argument to rtcSet() is ~~an~~ a trace instance handle and the second argument is the new setting for this instance.

---

Please replace the paragraph on page 53, lines 9 through 16, with the following amended paragraph:

---

a10  
ALG\_activate() initializes any scratch buffers and shared persistent memory using the persistent memory that is part of the algorithm's instance object. In preemptive environments, it saves all shared data memory used by this instance to a shadow memory so that it can be restored by ALG-deactivate() when this instance is deactivated. The

a10  
only argument to ALG\_activate() ~~ALG\_activate()~~ is an algorithm instance handle. This handle is used by the algorithm instance to identify the various buffers that must be initialized prior to execution of any processing functions. It has no return value.

---

Please replace the paragraph that begins on page 59, line 22, and ends on page 60, line 3, with the following amended paragraph:

---

a11  
The flow graph in Figure 20B illustrates the execution steps. In steps 2004 and 2005, client 2009 (a framework) queries the memory interface of algorithm module 2010 to get the memory usage requirements of an instance of the algorithm. Algorithm module 2010 responds to the memory allocation inquiry by returning the memory usage requirements for an instance of the algorithm. In step 2006, client 2009 allocates physical memory for algorithm instance 2011 based on the memory usage requirements it received. In step 2008a, algorithm module 2010 accepts the region of memory allocated by client 2009. Algorithm module 2010 then initializes the region of memory to create algorithm instance 2011 in step 2008b ~~2008-b~~. Steps 2004 – 2008b are is repeated for each algorithm module in the software program.

---

Please replace the paragraph on page 60, lines 23 through 29, with the following amended paragraph:

---

a12  
Alternatively, the framework may choose ~~choose~~ to execute the sequence of algAlloc() followed by algInit() each time it wishes to execute any of the algorithm specific functions of a given algorithm module. When the desired algorithm specific functions have completed execution, the framework can then execute algFree() to get the addresses and size of each memory block allocated to the algorithm module and free up that memory for use by another algorithm module.

---

Please replace the paragraph that begins on page 62, line 22, and ends on page 63, line 2, with the following amended paragraph:

a13  
The answer to the first question is determined by the implementation of the algorithm module. In an alternate embodiment, the memory interface is further extended to include the capability to adopt a model of using scratch memory versus persistent memory. This capability allows an algorithm module to divide its memory into two or more blocks ~~block~~. This subdivision of memory helps reduce fragmentation of physical memory and allows the framework to make more efficient use of physical memory since the area allocated to scratch memory can be shared among all algorithm modules in the application.

Please replace the paragraph on page 71, lines 14 through 19, with the following amended paragraph:

a14  
For increased flexibility in optimizing memory usage, the algorithm module should be extended to permit an algorithm instance to be activated, deactivated, and relocated by the framework at run-time. In this embodiment, this functionality is provided by implementing the functions algActivate(), algDeactivate(), and algMoved(). These functions are described ~~in~~ previously herein.

Please replace the paragraph that begins on page 71, line 26, and ends on page 72, line 9, with the following amended paragraph:

a15  
The present invention allows system designers to easily integrate algorithm modules from a variety of sources (e.g., third parties, customers, etc.). However, in system design, flexibility comes with a price. This price is paid in CPU cycles and memory space, both critical in all DSP systems, but perhaps most critical in a static system. A static system is one in which no memory allocation occurs at run-time. The worst-case amount of memory needed to execute the application can be determined when the software system is designed (at "design time") and that memory can be allocated statically among the algorithm modules in the software system. This static memory allocation is then initialized at run-time. There is no need for run-time memory allocation. Note that static systems may reuse memory. They will, for example, overlay

Q15

algorithm instances in the same physical memory when ~~its~~ it is known that the instances are never concurrently active.

---

Please replace the paragraph on page 78, lines 20 through 21, with the following amended paragraph:

---

Q16

Advantageously, object creation can be performed in a ~~non-real time~~ non-real-time manner. Object creation code can be overlaid and reused.

---